Chapter 17
Process and Control Documents

Introduction

This chapter contains documents and references for use during the project management processes, control functions, and quality review gates as discussed in this SDLC.

In This Chapter

Section A
SDLC & Process Improvement Recommendation (SPIR) Form

SDLC and PROCESS IMPROVEMENT RECOMMENDATION

Send To: Susan Coonrod/Regina Cohen
Technical Architecture Group (TAG)
Newington Data Center (NDC) Room 209
7681 Boston Blvd., Springfield, VA 22153

Date:

Contact Information		
Name:		Organization:
Project:	Room:	Phone:

Process/SDLC Chapter/Document Relating to Change

Suggestion/Change

Rationale

Section B
Functional Impact Areas Checklist

Functional Areas

The functional support areas listed below are suggestions for the PM's consideration concerning:

- Possible resources and support functional areas required by the project and
- Potential impacts of the project on other groups performing these functions

Resources Req'd?	Potential Impact?	Functional Description
System Operations-related Areas		
		Mainframe Support (includes hardware, software, processing, administration, etc.)
		Centralized UNIX Server Support (all aspects)
		Field UNIX Server Support (all aspects)
		Communications/Network (all data transfer aspects)
		Voice Communications (Telecom Lines/Dial tone)
		LAN Support (all aspects)
		Storage Management (DASD and Tape)
		System Services (CICS, JES2, MVS, MQM, etc.)
		Capacity Planning
		Electronic Data Interchange (EDI) Support
		Client/Server Deployment (field equipment, installation, training issues)
		Operations/Scheduling
		[Other functional areas as required]

Continued on next page

Functional Impact Areas Checklist, Continued

Functional Areas (continued)

Resources Req'd?	Potential Impact	Functional Description
		Data-related Areas
		Database Administration/Support
		Data Administration
		Data Quality Assurance
		Data Conversion
		Corporate Data Dictionary
		Standards/Oversight-related Areas
		Architecture Design/Oversight
		SDLC Compliance/Oversight
		User Interface Standards/Oversight (CICS, GUI)
		Post-Implementation Evaluation/Reviews
		Additional Functional Impact Areas
		Security
		Procurement
		Configuration Management [for development]
		Configuration Management [for acceptance testing and production]
		Testing [System Acceptance Testing]
		Training [User Training]
		User Documentation
		Help Desk Support / Customer Support

Continued on next page

Functional Impact Areas Checklist, Continued

Functional Areas (continued)

Resources Req'd?	Potential Impact	Functional Description
		[Other functional areas as required]
Interfaces with non-Customs Systems, Agencies (list):		
Other Related Projects (list interfaces):		
		Data Warehouse
Additional Contract Support (list type):		
Other Support (list type):		

See Volume I, Chapter 5, Section C, *Management Reviews/Oversight*, **Reporting Across OIT Organizations**, for further details on using this suggestion list.

Section C
COTS Product Evaluation Criteria

Keys to Success

The keys to the successful use of COTS are:

- A thorough evaluation of each candidate component or product by a competent team composed of users and technical personnel, and for enterprise level systems, a system integrato.
- A clear, comprehensive, well-defined set of User Requirement.

Reference: Volume II, Chapter 10, *Off-the-Shelf (COTS) Applications*

Section Overview

This section contains generic checklists of questions to be considered when evaluating an off-the-shelf product.

Additional topics and questions should be added to the evaluation criteria by the Product Evaluation Team depending on the domain and functionality of the product(s) under consideration.

In This Section

Requirements Evaluation

Criticality

User requirements must be carefully defined and reviewed to ensure that everyone involved in the evaluation process understands which requirements are:

- **Mandatory:** These requirements are mission-critical and non-negotiable.
- **Necessary but Negotiable:** These requirements must be satisfied, but they can be fulfilled by any one of several possible methods.
- **Nice to Have:** These requirements are not necessary to the support of the business area.

"Best" Practices

Three practices enhance the chances that a candidate package will fulfill its vendor promises:

- Each candidate software package should be evaluated by more than one person. If possible, every candidate package should be evaluated by every member of the team.
- Though evaluations should be independent, each evaluator should have a standardized list of the evaluation criteria.
- A single, final product requirements score should be reached by team consensus.

Continued on next page

Requirements Evaluation, Continued

Requirements Evaluation Procedure

The following procedure should be followed when evaluating the product against user requirements:

Step	Description
1	If possible, obtain a copy of the candidate component and all pertinent operating documentation from the vendor. **Warning:** All applicable copyright and licensing restrictions must be observed.
2	Verify that the software is virus-free and load it onto the target platform.
3	Score the candidate software's performance against each item in the user requirements list. **Note:** User requirements will have been thoroughly defined, documented, certified, and understood by all members of the Product Evaluation Team **before** product evaluation begins.

Cost Evaluation

Introduction

Cost factors can be one of the major advantages as well as one of the major disadvantages of choosing off-the-shelf software over building the product in house.

Costs associated with off-the-shelf software fall into three categories:

- Up-front costs
- On-going costs
- Upgrade/tailoring costs

Each of these is discussed in greater detail below.

"Best" Practices

Sound, basic business principles can do much to eliminate unpleasant surprises associated with product costs:

- Ensure that the vendor's representative has the authority to make binding cost quotes.

- Get all cost quotes in writing.

- If the product is to be purchased under a licensing agreement and/or contract, ask for assistance from the appropriate personnel within the Office of Finance.

Up-Front Costs

Definition: Up-Front Costs are those cost items which occur at the beginning of a product's life cycle within the Customs environment. They include (but are not necessarily limited to):

- Licensing and/or purchase costs

- Installation costs (including any tailoring) necessary to create interfaces with existing systems and/or databases

- Initial technical staff training and training for user trainers

- The availability and cost of source code

Continued on next page

Cost Evaluation, Continued

Up-Front Costs (continued)

- Costs associated with any supporting software required (i.e., a database or word processing package that acts as a front or back end to the candidate package)

On-going Costs

Definition: On-going costs are those costs which can reasonably be expected to recur on a regular basis during the product's life cycle. These include (but are not necessarily limited to):

- Periodic license renewals for both the candidate package and any supporting software packages, including the costs of adding additional users above and beyond those in the original agreement

- Fees for upgrades and maintenance

- Fees for technical support

- Fees for training and documentation updates

Tailoring Costs

If the vendor offers product tailoring, the Product Evaluation Team should ask the following questions:

- Will the addition of the tailoring help the product meet a mission critical need?

- What is the initial cost of the needed/proposed tailoring?

- If the product is upgraded or if there is a new release, will the tailoring have to be repeated? If so, what would be the cost?

Customization Costs

If this product's source code must be customized to meet mandatory requirements:

- How much will the initial customization cost?
- How much will it cost to repeat the customization for each new release?

Technical Evaluation

Introduction

The technical evaluation must address all aspects of how the product will function within the Customs architectural environment.

"Best" Practices

During the evaluation, assistance should be sought from all appropriate technical support personnel in order to obtain a clear understanding of:

- How the product will behave within the target environment
- What interfaces will be necessary to ensure successful integration

Technical Areas and Sources

The following technical areas should be investigated for each product:

For This Issue...	Consult ...
Interfaces to Current Systems and/or Other Components	• Users and/or Documentation on the System to be Replaced • The Business Sponsor/Process Owner • Vendor/Integrator
Dependencies on Current Systems	• Business Sponsor/Process Owner • Users of and/or Documentation on the system to be replaced
Performance Issues	• Performance Reporting Team, OIT • Technical Architecture Group
Equipment Requirements	• Technical Architecture Group • Vendor's Documentation • National Logistics Center (if it is anticipated that the component will require replacement or augmentation of current equipment)

Continued on next page

Technical Evaluation, Continued

Technical Areas and Sources (continued)

For This Issue...	Consult ...
Systems and Data Security	• AIS Security Team, OIT • Vendor's certification
Capacity Issues	• Capacity Planning Team, OIT
Product Quality/Reliability	• References supplied by the vendor • Colleagues in other government agencies • Gartner group or other industry "consumer group"
Database	• Data Administration Team, OIT • Database Management Team, OIT
Year 2000 Compliance	• Y2K Program Office, OIT • Vendor certification

Testing or Benchmarking

During this evaluation, products should also be compared as to how well (or whether) they meet the acceptance test criteria which have been developed based on the user requirements.

In some cases, this comparison will involve testing an evaluation copy of each product and benchmarking its results in a controlled environment.

Vendor Evaluation

Introduction

By selecting an off-the-shelf package, the Customs Service will be establishing a long-term relationship with the vendor. Therefore, the vendor's reputation, stability, and past performance should be evaluated in addition to the product's performance and stability.

Support Offered or Provided

In evaluating the support offered by a vendor, the Product Evaluation Team may wish to ask the following questions among others:

- What types of support is the vendor offering? Onsite? Telephone?
- What are the costs for these support services? What are up-front costs? Is there a service-call fee if there are problems later?
- Are there any other kinds of support offered that were not evaluated under "Costs" or "Technical Factors"?

Vendor History in the Marketplace

Vendor reliability must be evaluated. This includes evaluating:

- How long has this software vendor/producer been in business?
- If it is a publicly traded company, can we obtain a copy of the last annual report? Is the company in sound financial condition?
- Can the vendor provide references in other organizations — preferably other government agencies?
- Can the vendor provide ratings/certifications by recognized industry groups?

Integrator Evaluation

Introduction

If a third-party integrator (i.e., someone separate from both the software vendor and Customs) is proposed, the services and experience of the integrator should be evaluated separately from the vendor.

Support Offered or Provided

In evaluating the support offered by an integrator, the Product Evaluation Team may wish to ask the following questions among others:

- What types of support is the integrator offering? Onsite? Telephone?
- What are the costs for these support services? What are up-front costs? Is there a service-call fee if there are problems later?
- Where is the integrator located? Can they come onsite rapidly if a component or interface fails?
- Are there any other kinds of support offered that were not evaluated under “Costs” or “Technical Factors”?

Integrator History

Integrator competence must be evaluated. This includes evaluating:

- How long has this software integrator been in business?
- How many times has he/she performed this type of component integration? What were the outcomes?
- Can the integrator provide references in other organizations — preferably other government agencies?
- Can the integrator meet security requirements? Integrator tasks require access to a Customs facility. Can the integrator (and/or his staff) meet clearance requirements?

Section D
Software Process Summary Questionnaires

Introduction

The questions outlined in this section provide a structured starter set to:

- Discuss and document project experiences and lessons learned
- Assist in initial project planning and review/comparison of lessons learned

The resulting Lessons Learned Report can be tailored to:

- Provide additional information on what worked (or was “most effective”) and what didn’t work (or was “most risky”)
- Make recommendations for the benefit of future maintenance and enhancement projects
- Suggest improvements to the processes of other current and future projects

In This Section

Lessons Learned/Postmortem Questions

Introduction

Please consider these 14 questions to assess the process used on this project, and, in contrast to prior practices, what was better and/or what was worse?

1. Communication and Information

- What was done to ensure communication and definition of the project:
  - Internally?
  - With interdependencies and support organizations?
  - With clients?

2. Software Life Cycle

- What was the overall sequence of events or milestones for the project?

3. Program Statement and Requirements

- How did you know “what was in and what was out” for requirements and features?
- If requirements changed, how was this communicated?

4. Requirements Verification and Release Criteria

- How did you determine a requirement was met or a feature was complete?
- How did you determine the product was ready?

5. Design and Design Verification

- How were product functions designed?
  - Prototype?
  - Formal designs?
- How were the designs evaluated?
  - Reviews?
  - Inspections?
  - Approvals?
- How were design errors identified and removed?

Continued on next page

Lessons Learned/Postmortem Questions, Continued

6. Code and Code Verification

- How was the software code developed?
- How was the code evaluated, and how were defects discovered?
  - Inspections?
  - Unit test?
- If multiple evaluation methods were used, what selection criteria were used?
- How were software defects identified and removed?

7. Integration and Qualification

- How was the software integrated and tested?
- How was the testing planned?
  - Tests for integration?
  - System?
  - User?
  - Security?
- How were defects handled?
  - Tracked to closure?
  - Deciding who gets to fix the defect?

8. Preparation for Support

- How were regression tests developed and maintained?
- How were test results reported and to whom?
- What information is available to those who will maintain or interface with the product in the future:
  - Training materials?
  - Designs?
  - Documentation?

Continued on next page

Lessons Learned/Postmortem Questions, Continued

9. Packaging and Delivery

- How was the user documentation designed and integrated with the product?
- How was the product packaged and delivered for release?

10. Project Planning and Tracking

- How were the project tasks determined, estimated, communicated, and tracked?
- How was status evaluated (such as "schedule on track" or "behind schedule")?
- How were impacts of changes accounted for on the project (for example, requirements changes, defect levels, staffing changes)?

11. Configuration Management

- How was the code managed and controlled?
  - ▸ During:
    -- Development?
    -- Testing?
    -- For production?
  - ▸ For version control:
    -- Tracking the "current levels" or current files or patches?
    -- Patch reviews or approvals?
    -- Source code control tools?

Continued on next page

Lessons Learned/Postmortem Questions, Continued

12. Project Coordination and Risk Management

- How were commitments managed and communicated?
    - Between project and users/clients?
    - Between project and support organizations?
- How were risks managed and communicated?
- How were corrective actions identified, planned, and tracked?
- How were the project activities reviewed or verified?
- How were the needs for tools or training identified, planned, and tracked?

13. Process Effectiveness

- What was most effective about doing the project in this way?
- What would you recommend to keep or change about the process based on this?

14. Process Risk

- What was most risky about doing the project in this way?
- What would you recommend to keep or change about the process based on this?

New Project Planning Questions

Introduction

Please consider these 14 questions to determine the process to use on this project. Insights may be obtained by reviewing the responses documented in the Lessons Learned Report of similar prior projects.

1. Communication and Information

- What will be done to ensure communication and definition of the project:
  - Internally?
  - With interdependencies and support organizations?
  - With clients?

2. Software Life Cycle

- What will the overall sequence of events or milestones for the project be?

3. Program Statement and Requirements

- How will you know “what is in and what is out” for requirements and features?
- If requirements changed, how will this be communicated?

4. Requirements Verification and Release Criteria

- How will you determine a requirement has been met or a feature has been completed?
- How will you determine the product is ready?

5. Design and Design Verification

- How will product functions be designed?
  - Prototype?
  - Formal designs?
- How will the designs be evaluated?
  - Reviews?
  - Inspections?
  - Approvals?
- How will design errors be identified and removed?

Continued on next page

New Project Planning Questions, Continued

6. Code and Code Verification

- How will the software code be developed?
- How will the code be evaluated, and how will defects be discovered?
  - Inspections?
  - Unit test?
- If multiple evaluation methods are used, what selection criteria will be used?
- How will software defects be identified and removed?

7. Integration and Qualification

- How will the software be integrated and tested?
- How will the testing be planned?
- How were defects handled:
  - Tracked to closure?
  - Deciding who gets to fix the defect?

8. Preparation for Support

- How will regression tests be developed and maintained?
- How will test results be reported and to whom?
- What information will be available to those who will maintain or interface with the product in the future?
  - Training materials?
  - Designs?
  - Documentation?

9. Packaging and Delivery

- How will the user documentation be designed and integrated with the product?
- How will the product be packaged and delivered for release?

Continued on next page

New Project Planning Questions, Continued

10. Project Planning and Tracking

- How will the project tasks be:
  - Determined?
  - Estimated?
  - Communicated?
  - Tracked?
- How will status be evaluated (such as “schedule on track” or “behind schedule”)?
- How will impacts of changes be accounted for on the project (for example, requirements changes, defect levels, staffing changes)?

11. Configuration Management

- How will the code be managed and controlled for version control?
  - Tracking the “current levels” or current files or patches?
  - Patch reviews or approvals?
  - Source code control tools?

12. Project Coordination and Risk Management

- How will commitments be managed and communicated?
  - Between project and users/clients?
  - Between project and support organizations?
- How will risks be managed and communicated?
- How will corrective actions be identified, planned, and tracked?
- How will project activities be reviewed or verified?
- How will the needs for tools or training be identified, planned, and tracked?

13. Process Effectiveness

- What do you expect to be most effective about doing the project in this way?

14. Process Risk

- What do you expect to be most risky about doing the project in this way?

Section E
Guidelines for Specific Core Project Metrics

Overview

The guidance presented below is designed to help the manager create the required indicators. Each graph should show the previous 12 months of data points (as available).

Although guidance is presented here, it is understood that each project and support organization has its own unique processes.

Documentation

To address these differences and ensure accurate interpretation of results, it is suggested that each project/organization document their individual interpretations/definitions of each graph.

Information on Indicators

For each indicator discussed below, the following information will be provided:

- Definition of the indicator
- Some questions addressed by the indicator
- Description of the data to be graphed
- Source of data
- Sample graph/display

In This Section

Milestone Performance

Definition

Significant **Milestone Dates** (based on Project Plan) are listed and presented in a tabular or Gantt Chart report format. This report indicates the original plan (baseline) date, the current revised date, and the actual completion date.

Milestones To Be Included

Milestone dates to be identified for a project must include the planned dates for:

- Requirements Certification,
- Design Completion/Acceptance (e.g., Critical Design Review [CDR]),
- Turnover to System Acceptance Test (SAT) Team,
- System/Release Implementation, and
- Critical OIT Support Organization Deliverables.

Questions Addressed

1. What major achievements has the project completed?

2. Is the current schedule realistic?

3. How often has the schedule changed?

4. How many activities are concurrently scheduled?

5. Are there significant delays between completion of a deliverable and its review/approval?

Data Description

Dates must be clearly defined for each milestone. A number of dates can be selected and shown for each milestone to best represent the project's status.

To illustrate this definitional clarity, the numbers used to identify specific date definitions in the Examples below are again referenced in the Usage Example and Sample Report Display to follow.

Examples of Relevant Date Definitions:

1. Internal review complete
2. Formal review with customer/user complete

Continued on next page

Milestone Performance, Continued

Data Descriptions (continued)

Examples of Relevant Date Definitions: (continued)

3. Sign-off by customer/user
4. All high-priority action items closed
5. All action items closed
6. Product of activity/phase placed under configuration management
7. Inspection of product signed off by QA
8. QA sign-off
9. Management sign-off

Usage Example

The report will show all relevant dates for each specified milestone (see the Sample Report Display below).

Often, the example date definition items 2, 3, and 6 are all significant milestones for one or more deliverables (e.g., for specification and design documents).

Other milestones may only need to report the completion of one of these defined dates.

Reference: See the Sample Schedule Checklist at the end of this section to help select and define specific milestones.

Sources of Data

- Project Plan/Schedule
- Project status reviews and reports

Continued on next page

Milestone Performance, Continued

Sample Report Display

Project: ____________ System/Release: ______ Period Ending: _______
Originator: ____________ Phone: ___________

Milestones, Reviews, and Audits Completed	Planned Date	Chg?	Revised Date	Actual Completion
Requirements Specification Review	▨	▨	▨	▨
2 - Formal review with customer				
3 - Sign-off by customer				
6 - Product under CM				
Critical Design Review	▨	▨	▨	▨
2 - Formal review/turnover meeting				
3 - Sign-off by customer				
6 - Product under CM				
Code Complete	▨	▨	▨	▨
1 - Internal review complete				
Unit Test Complete	▨	▨	▨	▨
5 - All action items closed				
7 - Inspection signed-off by QA				
Test Readiness Review	▨	▨	▨	▨
8 - QA sign-off				
Delivery and Installation	▨	▨	▨	▨
3 - Sign-off by customer				
6 - Product under CM				
8 - QA sign-off				
9 - Management sign-off				

Enter a check in the “Chg?” column if Planned or Revised date has changed since last reporting.

Milestone Percent Complete

Definition

Milestone Percent Complete is calculated by specifically identified subcomponents/units to be completed by specified dates within a milestone.

No partial credit is given. These mini-milestones or 'inchstones' are not defined as complete until all items/units included within them are complete.

If it is not fully complete, it is not included within the Milestone % Complete.

Questions Addressed

1. How are specific critical activities and products progressing?

2. Is the project meeting scheduled milestones?

3. Is the planned rate of progress realistic?

Description of Data

The project-specific units to be completed as part of an inchstone and their specified completion criteria are the same as those used for the Progress Indicator defined below.

Units or components may be defined differently for each project and phase/task. These can be programs, modules, objects, interfaces, screens, reports, problem reports, pages, tables, or other measurable product structure.

These specified units are used for estimating and tracking to define progress within the overall activity/milestone. As the estimated number of units included in a milestone is revised, the Milestone % Complete amount will also change and may decrease.

Continued on next page

Milestone Percent Complete, Continued

Usage Example

A Components Designed measure is defined to count the number of software components that have completed preliminary or detailed design. The total number of components is estimated, and the number to be completed over time are defined for comparison with the actual number completed. Comparing planned vs actual components (both total number and number to be completed) is very effective for assessing design progress.

When analyzing results of this metric early in the design phase, planning changes would be expected as the design matures. Later in the process, an increase in the planned number of components will cause the % Complete amount to decrease and can be an indication of unplanned or excessive growth.

Sources of Data

- Project Plan/Schedule
- Project status reviews and reports

Sample Display

A '% Complete' column can be:

- added to the Milestone Dates report

AND/OR

- maintained and printed as part of a Project Schedule/Gantt Chart

for each task/activity. Relevant units used must be clearly defined on the report.

Progress Indicators

Definition

The Progress Indicator compares the **number of items/units** which are expected/estimated to be completed over time against the number of specific units actually completed. The items/units chosen are specifically predefined for the project or support organization/team within each phase/time period to be performed.

Every functional area within a project or Support Organization must define measures to allow progress to be tracked on a weekly or biweekly basis. Longer-term milestones are not sufficient to allow for schedule slippage to be noticed and corrected before it becomes a major source of overruns and replanning.

Questions Addressed

1. Is the project on schedule?
2. Which components/functions are behind/ahead of schedule?
3. Is the project within budget?/Is the planned rate of progress realistic?
4. Are revisions to the schedule/cost following established procedures when the actual results and performance data indicate replanning is necessary?

Description of Data

Depending on the phase or task, the types of units/items which can be predefined, estimated/planned, and compared with their actual numbers during the project are, for example:

- Number of Components (e.g., modules, database tables, objects):
  - Designed
  - Implemented
  - Tested
- Number of Requirements:
  - Baselined
  - Tested
- Number of Test Cases Completed

Continued on next page

Progress Indicators, Continued

Description of Data (continued)

- Number of Reviews Completed
- Number of Problem Reports Resolved
- Number of Changes Implemented
- Number of Units Installed
- Number of Pages Written or Modules/Functions Documented

The units identified above can be defined to indicate progress by Support Organizations on their deliverables and milestones, in addition to the more traditional units which indicate software development progress by phase.

Notes

- This comparison of planned vs actual accomplishment provides management with an overview of the value of work completed to date, indicates whether corrective actions are required, and directly correlates with schedule and cost variances on the project.
- Objective criteria are key to clearly defining when an item/unit is considered complete.

  Reference: See Part B of the Sample Schedule Checklist at the end of this section.

Examples of Work Unit Completion Criteria

- Passed peer review
- Entered under configuration control
- All relevant action items closed
- All test cases completed with no defects
- Signed off by first-line supervisor

Sources of Data

- Cost/schedule status reports
- Project status meetings/reviews
- Project configuration management and control databases

Continued on next page

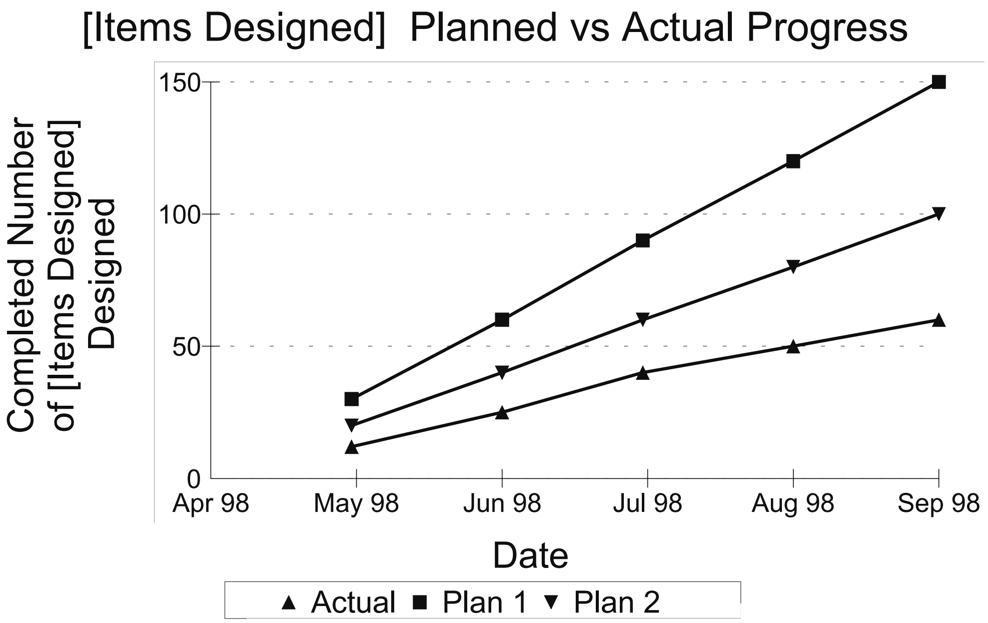
Progress Indicators, Continued

Sample Trend Graph Display

For the most visibility and ease of use, the comparison of planned vs actual accomplishment for selected units/indicators is recommended to be displayed over time, using trend graphs. Units used must be clearly defined.

The sample shown indicates progress (planned vs actual) by components (e.g., modules, objects, database tables, document pages/chapters, installations, etc.) which have completed Design. The completion criteria should be clearly defined (e.g., in this example the items designed have passed peer reviews).

This example shows two plans (e.g., the original plan and a revised plan) and the actual number of completely designed components over time.



Effort Indicator

Definition

Number of **Staff Hours** Spent, cumulative over time; preferably accumulated by subtasks within each phase.

This Effort indicator usually correlates directly with cost, and can also be used to address other common issues including schedule progress and development performance.

At minimum, for cost comparison/replanning, the actual level of effort spent must be accumulated (or rolled up) into the same groupings as was used in estimating the level of effort required.

Questions Addressed

1. Is effort being expended according to plan?

2. Are certain tasks/activities taking more/less effort than expected?

3. Is the effort profile/budget realistic?

Description of Data

Accumulate the actual number of staff hours spent on each task/subtask for comparison with the planned number of hours estimated for this work. Trend data provides an early indication of potential schedule and budget overruns.

Government FTE hours and Contractor Staff Hours may be accumulated separately for comparison with the estimated level of effort and budgeted costs.

Usage Example

Staff Hours accumulated by Subtask/Activity within the Programming Phase for a Developer could include the hours spent:

- Coding a module/program
- Testing the module
- Documenting the module
- Participating in Reviews/Code Walkthroughs

This accumulated level of effort metric also permits the collection of historical Customs data on the actual time spent on specific tasks so that future planning efforts for later releases and similar projects will be more accurate.

Continued on next page

Effort Indicator, Continued

Source of Data

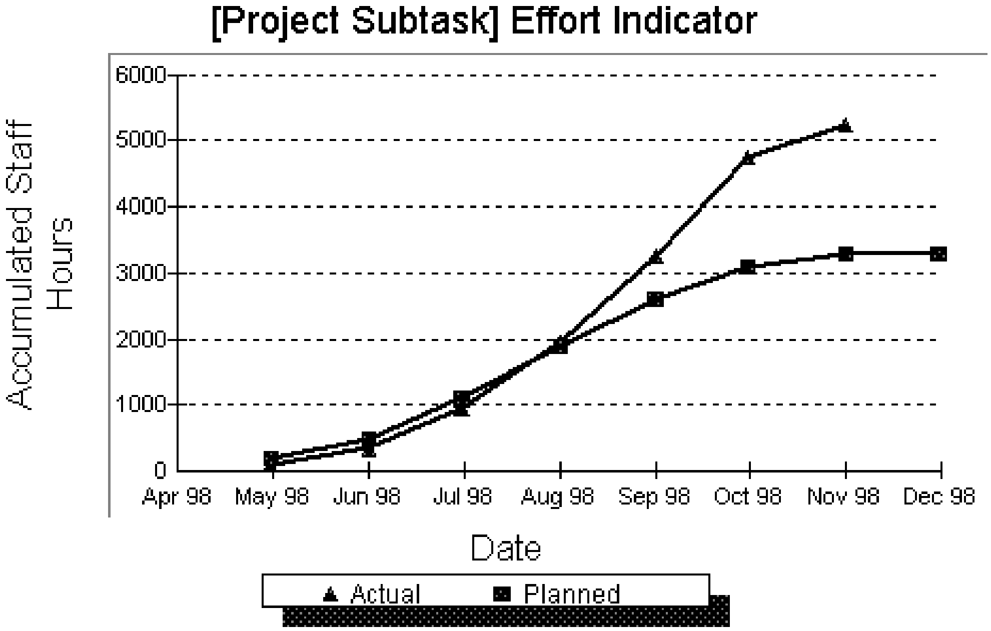
Invoices, Project Status Reports.

The project/support organization can accumulate actual level of effort data on a daily or weekly basis by having each team member:

- Add their actual hours for each task/activity into an accumulator field of a project-specific defined database or spreadsheet
  Or
- Provide their actual hours data to team leads or a metrics analyst for entry of group totals into MS Project or a project hours datamart

In the future, hours for pre-defined activities within each project may be extracted from a standardized Customs accounting system database or datamart.

Sample Trend Graph Display



Staffing Indicator

Definition

Shows the number of **Actual Staff** assigned to the project over time and in comparison with the planned number of staff over the same period.

As appropriate to evaluate impact on the project, this information can be collected and displayed broken down by type of position filled and/or status of the potential [actual, but not yet functional] staff's background clearance investigation.

Questions Addressed

1. Are sufficient development resources available and applied?
2. Is project progress consistent with the level of effort/resources expended?
3. Are qualified staff assigned according to plan?
4. Are certain activities or functions taking more staff than expected?
5. If there are overruns or underruns is replanning necessary?

Description of Data

The Staff Level measure counts the total number of personnel assigned to specific activities over time and compares that with the estimates used to schedule and cost the project. Data is collected in equivalent person-counts.

This measure is used to determine if sufficient personnel are available to perform the task and can also provide an early indication of possible schedule slips and cost overruns or underruns.

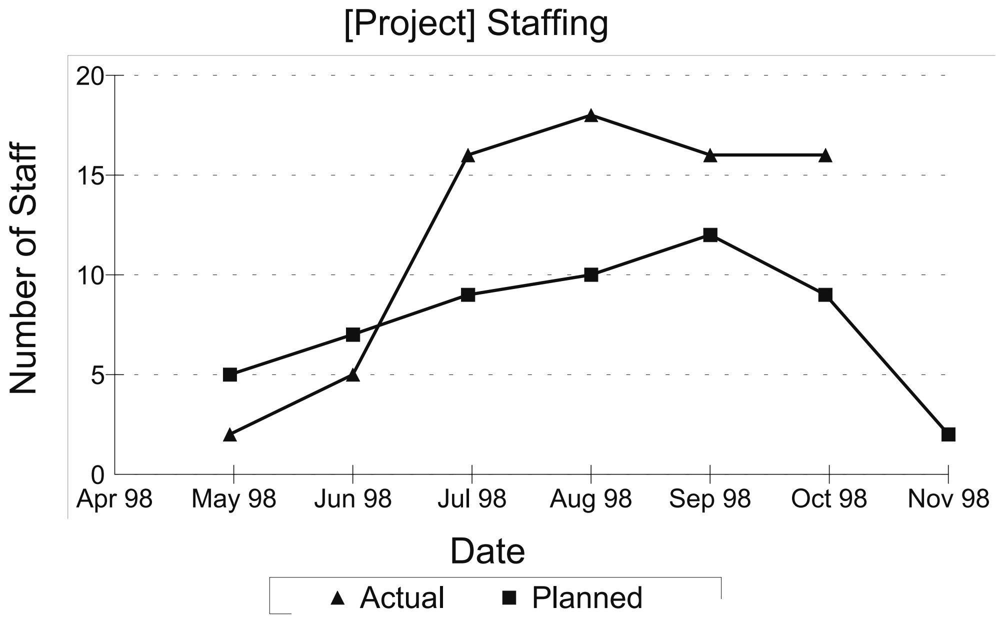
Source of Data

- Project status meetings/reports
- Invoices for contract labor

Continued on next page

Staffing Indicator, Continued

Sample Trend Graph Display



Alternate Sample Report Display

Project: _____________ System/Release: ______ Period Ending: _______
Originator: _____________ Phone: ___________

Labor Category (Examples)	Staffing Levels Planned	Actual Staffing Levels (FTEs)		
		On Board	Cleared	w/LAN Access
Designers				
Developers				
Technical Writers				
Technical Support				

Product Size and Stability

Definition

Work Products Size metrics quantify the physical size of each of a project's or support organization's work products. Size stability is indicated by comparing the planned size vs the actual size over time. Product size is a critical factor for estimating development schedule and cost.

Trends in this area can be used to measure growth and stability of the work, progress (comparing estimated size to actual size), productivity, and required effort and rework. It also provides insight into the amount and frequency of change which is especially critical late in the development.

Questions Addressed

1. How accurate was the size estimate(s) that the schedule and effort plans were based on? How reliable are ongoing size measurements?

2. How much has the size changed? What is the trend in code size? In what components have changes occurred?

3. Has the size allocated to each incremental build changed? Is functionality slipping to later builds?

4. Are resources (e.g., money, people, equipment) in place to absorb any growth in size?

Description of Data

The specific units to be used for the size estimates are to be defined by the individual project/organization and can include metrics for various types of work products. The same type of units are compared for trend analysis of estimates vs. actuals on a specific project. The unit(s) used must be specified on all reports.

Examples of Work Product Size Measures

- Lines of code
- Function points
- Number of objects
- Number of components
- Number of documentation pages
- Database size

Continued on next page

Product Size and Stability, Continued

Examples of Work Product Size Measures (continued)

It is important to be clear about the definition of the selected unit of measure for the project.

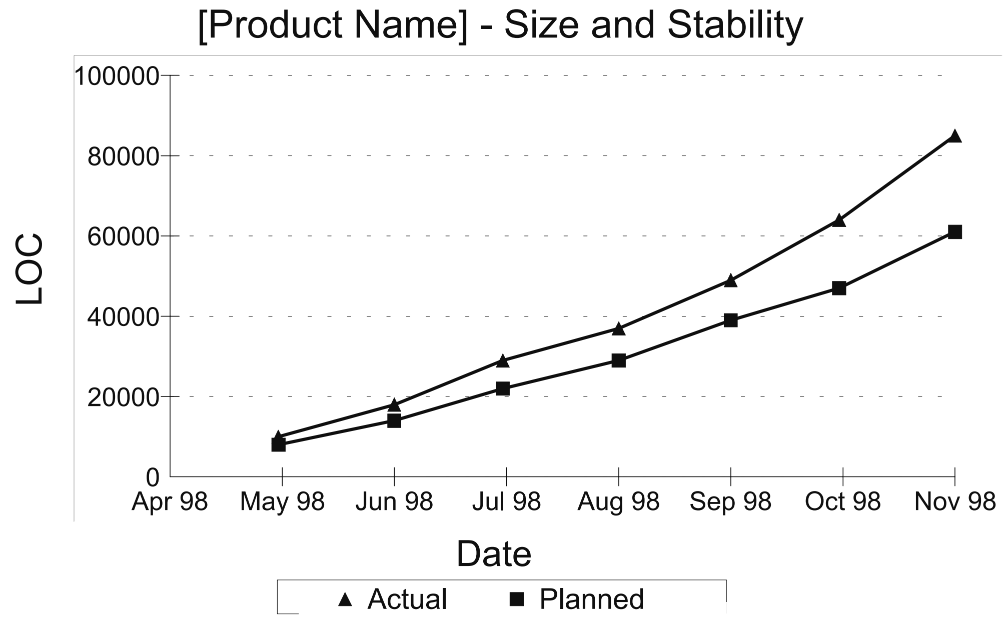
Example: Does 'lines of code' include comments, data declarations, or blank lines; do we count lines deleted or modified as well as those added?

Source of Data

- Automated counts from the Project CM database
- Monthly reviews

Sample Trend Graph Display

The example shown provides the estimated vs actual size for software modules in Lines of Code. Similar graphs could be created showing other measures (e.g., Number of Objects/Screens/Function Points) and/or for other work products (e.g., documentation, database tables).



Functional Size and Stability

Definition

The **Requirements** measure counts the number of requirements in the software and interface specifications, and the number of these requirements that are added, modified, or deleted over time.

Tracking the number of requirements addresses the stability of the system's functionality and capability. Requirement additions/modifications should decrease as the development progresses (e.g., after preliminary design review).

Questions Addressed

1. Are requirements and functionality changing? Are requirement changes consistent with the life cycle phase of the project?

2. Have the requirements allocated to each incremental build changed? Are requirements being deferred to later builds?

3. How much has functionality changed? What components have been affected the most?

4. What is the trend in requirement changes/product stability and how will that affect rework effort and schedule?

Description of Data

Count changes to requirements against a baseline that is under formal configuration control (e.g., after Requirements Certification).

Two metrics are obtained over time:

- the **Number of Total Requirements** and

- the **Number of Changes to Requirements** where
  Changes to Requirements = Added Requirements + Modified Requirements + Deleted Requirements

Both stated and derived requirements may be included.

Continued on next page

Functional Size and Stability, Continued

Usage Examples

- Even before baselining, if there is a significant difference between the planned total number of requirements and the actual number, this can indicate problems in the estimated size, schedule, and possibly cost of the project.

- If the number of changes to requirements is increasing after the requirements have been baselined, this can indicate:

  - Increasing need for rework and schedule/cost overruns

  - A need for improvement in the requirements gathering and analysis process used

    Reference: Volume I, Chapter 4, Section E, *Requirements Change Control/Impact Analysis*, definition of Changed Requirements.

Source of Data

- Requirements documents/Certification Baseline
- Internal Configuration Control Board data and Approved Changes
- Project Status reports, and/or project review packages

It is preferable to obtain counts as an automated extract from the project's Requirements database or traceability matrix, which must be maintained in a timely manner.

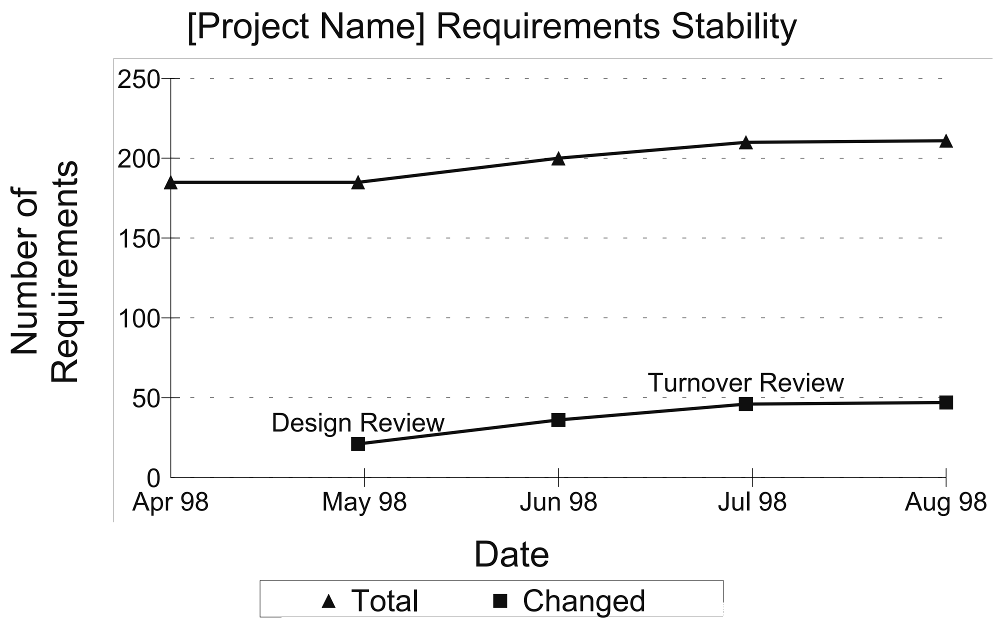
Sample Trend Graph/Display

It is suggested that this stability indicator be displayed using trend lines over time on one chart, as in the sample below. Alternatively, two trend graphs can be developed, one for each metric.

Continued on next page

Functional Size and Stability, Continued

Sample Trend Graph/Display (continued)



Product Quality Indicators

Definition

The Product Quality Indicators consist of the **Number of Saves** (that is, the Number of Discrepancies found against requirements and specifications), collected by:

- Phase (e.g., requirements, design, development, and implementation)
- Component (e.g., unit, module, table, object)
- Severity (e.g., priority or class)
- Status (e.g., open/closed or identified/analyzed/in work/resolved)

This measure is usually available during integration and testing. It is beneficial to begin problem tracking earlier, however, and also to consider the number of saves found relative to the amount of discovery activity on the project (e.g., number of inspections and reviews performed, amount of testing, etc.).

Questions Addressed

1. How many (critical) problem/save reports have been written? What is the trend in volume of discrepancies?

2. Do problem/save report arrival and closure rates support the scheduled completion date of integration and test?

3. How many problem/save reports are open? What are their priorities? Are the problems that are more difficult to fix being deferred?

4. Are resources being allocated to solving the backlog of defects?

5. Is the product good enough for delivery to the user?

Description of Data

Discrepancies (also known as defects, errors, or bugs) are found during all portions of the life cycle. The earlier that discrepancies are found in the life cycle, the less effort will be needed to fix the item. This will save the project later and larger adverse impacts on both schedule and cost.

This measure quantifies the number, status, and priority of discrepancies reported. The quantity of problems reported reflects the amount of development rework (quality).

Continued on next page

Product Quality Indicators, Continued

Description of Data (continued)

Arrival rates can indicate product maturity (a decrease should occur as testing is completed). Closure rates are an indication of progress and can be used to predict test completion.

Enhancements or improvements may also be determined at these times; these are treated as Changes. Enhancements/improvements should be evaluated for impact both on the work product under development and on the project's schedule and costs.

Sources of Data

- Product Acceptance/Review/Inspection notes
- Error Reports (e.g., TPRs, OPRs)
- Automated extract from Project Saves/Error status database

Notes on Reporting

Two Product Quality Indicator graphs, at minimum, should be developed based on the data collected:

- **Defect Profile/Saves Indicator** shows the number of total, open, and closed Saves over time (trend graph).

- **Discrepancies Found by Severity** illustrates the number of open discrepancies by priority or class. This metric may be reported as a bar graph showing the total number of problem/save reports and the number open in each priority as of the reporting date.

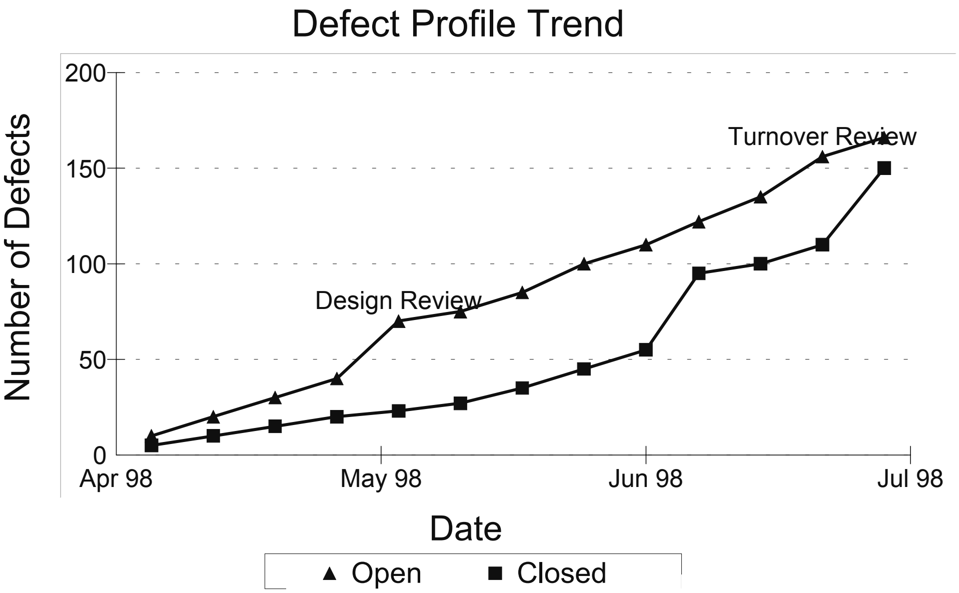
  Example of Priority/Severity Definitions (project-specific):

  - 1 = Critical
  - 2 = Complex
  - 3 = Simple
  - 4 = Cosmetic

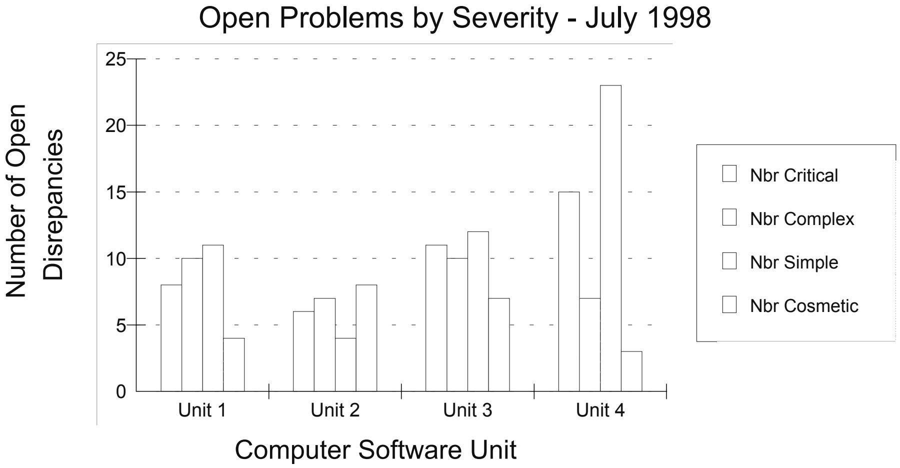
Continued on next page

Product Quality Indicators, Continued

Sample Defect Profile Trend Graph/Display



Sample Open Problems Severity Graph/Display



Charting Techniques Summary

Technique Checklist for Effective Graphing

Some guidelines for developing effective graphs include the following:

- Provide a descriptive title which identifies the indicator name, type of data, and component or unit (if applicable) represented by the graph.
- Show an as-of line or date represented by this data; many graphs will show plans or projections beyond the as-of date.
- Axis labels should include type of units and scale markers (e.g., dates or counts).
- Label line, bar, and data points directly on the figure or in a key/legend box on each page.
- Use similar conventions for all reports. For example, always use solid boxes for actuals and open boxes for plans.
- Use contrasting styles for lines, bars, and data points that represent different data. **Caution:** Take care not to depend only on color to distinguish between data types.
- Provide indicators of major milestones when showing time trends.
- Identify the source of the data (including version number) if appropriate.
- Graphs should show the previous 12 months of data points (as available).
- Use connect-the-dots technique for the display line rather than curve-fitting to show trends.
- Adjust the horizontal axis to show the expected range of the data to be plotted.

Continued on next page

Charting Techniques Summary, Continued

Technique Checklist for Effective Graphing

- Label significant events and trends in the data.
- Use the same axes on both when comparing two graphs.

  Example: It may prove useful to overlay a milestone Gantt chart onto the staffing profile planned across the same 12-month period to determine whether the allocation of staffing over time matches the schedule's needs.

Sample Schedule Checklist

Inroduction

The following pages provide an example of a two-part checklist used to help select and clearly define specific milestone date and progress completion points for the Milestone and Progress Indicators.

Note: These pages are provided as examples for reference from the SEI metrics publications, particularly CMU/SEI-92-TR-21.

Part A: Date Information

Originator:_____________________________Date:______________

Project will record planned dates: _____ Yes _____No
If Yes, reporting frequency: _____Weekly _____Monthly
Other: __________

Project will record actual dates: _____Yes _____No
If Yes, reporting frequency: _____Weekly ____Monthly
Other: __________

Number of builds: ____________

	Include	Exclude	Repeat Each Build	Relevant Dates Reported*
Milestones Reviews and Audits*				
System-Level				
System Requirements Review				
Technical Architecture Review				
System Design Review				
CSCI-Level				
Software specification review				
Preliminary design review				
Critical design review				
Code complete				
Unit test complete				

Continued on next page

Sample Schedule Checklist, Continued

Part A: Date Information (continued)

	Include	Exclude	Repeat Each Build	Relevant Dates Reported*
CSC integration and test complete				
Test readiness review				
CSCI functional and physical configuration audits				
System-Level				
Preliminary qualification test				
Formal qualification test				
Delivery and installation				
Other system-level: e.g., Delivery to prime contractor				
Deliverable Products**				
System-Level				
Preliminary system specification				
System/segment design document				
Preliminary interface requirements spec.				
Interface design document				
Software development plan				
Software test plan				
Software product specification(s)				
Software user's manual				
Software programmer's manual				
Firmware support manual				
Computer resources integrated support doc.				
Computer system operator's manual				

Continued on next page

Sample Schedule Checklist, Continued

Part A: Date Information (continued)

	Include	Exclude	Repeat	Relevant Dates Reported**
CSCI-Level				
Preliminary software requirements spec(s)				
Software requirements specification(s)				
Software preliminary design document(s)				
Software (detailed) design document(s)				
Software test description(s) (cases)				
Software test description(s) (procedures)				
Software test report(s)				
Source code				
Software development files				
Version description document(s)				

***Key to indicate "Relevant Dates Reported" for Reviews and Audits**:

1 - Internal review complete
2 - Formal review with customer complete
3 - Sign-off by customer
4 - All high-priority action items closed
5 - All action items closed
6 - Product of activity/phase placed under CM
7 - Inspection of product signed-off by QA
8 - QA sign-off
9 - Management sign-off
10 - ______________________________
11 - ______________________________

****Key to indicate "Relevant Dates Reported" for Deliverable Products**:

1 - Product under configuration control
2 - Internal delivery
3 - Delivery to customer
4 - Customer comments received
5 - Changes incorporated
6 - Sign-off by customer
7 - ______________________________
8 - ______________________________

Continued on next page

Sample Schedule Checklist, Continued

Part B: Progress/Status Information

Project will record planned progress: _____ Yes _____No
If Yes, reporting frequency: ______Weekly _____Monthly
Other:___________

Project will record actual progress: _____Yes _____No
If Yes, reporting frequency: ______Weekly _____Monthly
Other: __________

Activities	Work Units	Work Unit Completion Criterion*
CSCI Requirements Analysis	Requirements documented or specified	
CSCI Preliminary Design	Requirements allocated to CSCs	
	CSCs designed	
CSCI Detailed Design	CSUs designed	
CSU Coding and Unit Testing	Lines coded	
	Lines unit tested	
	Number CSUs coded	
	Number CSUs unit tested	
	Number lines unit tested	
CSCI Integration	Number of CSUs integrated	
	Number of lines integrated	
CSCI Testing	Number of tests passed	

***Keys to indicate "Work Unit Completion Criterion"**:

1 - None specified
2 - Peer Review held
3 - Engineering Review held
4 - QA sign-off
5 - Manager or supervisor sign-off
6 - Inspected
7 - Configuration controlled
8 - Entry in employee status report
9 - No known deficiencies
10 - Reviewed by customer
11 - All relevant action items closed
12 - ______________________________

Section F
Quality Assurance Checklists

Checklists

Checklists are useful to assist in performing verification functions. Each project may develop checklists specific to its needs.

The audit checklists and the deliverable inspection checklists provided are samples from Customs projects which can be adapted for use by other projects. Following these examples are SDLC Phase Checklists designed to assist the Project Manager in overall life cycle phase reviews of the project.

Types of Checklists

The following three types of checklists are used at Customs:

- Audit Checklists for ensuring that the project is ready to move to the next life cycle phase.
- Inspection Checklists for use in reviewing specific work products.
- SDLC Phase Checklists for preparing for Quality Gate review activities between each phase.

  Note: At each phase, all prior phase checklists should also be scanned to ensure that no updates are needed. Additional items can be added to each phase checklist depending on the project and its deliverables.

Reference: Volume II, Chapter 7, *Traditional Waterfall Life Cycle*

Updates

The samples below may be modified and enhanced. Please forward suggestions to the SDLC Team for potential incorporation in future versions of these aids.

In This Section

Audit Checklists

Introduction

This example illustrates a tailored life cycle flow used on a project as a quick reference checklist for the System Development Team, and as an audit checklist summary for Project QA phase validation activities.

Initiation Phase	
Goals	• Economic, technical, and operational feasibility studies • Obtain concept approval • Develop security safeguards • Obtain project approval
Product to be Prepared	• Project Summary • Project Concept • Project Management Plan • Project Implementation Plan • IT Funding Request Worksheet
Definition Phase	
Goals	• Verify complete and accurate specification for each of the following: ▸ Software Functions ▸ Inputs and Outputs ▸ States and Modes ▸ Response Time Requirements ▸ Interfaces • Ensure specifications are included for error detection and recovery • Ensure specifications for reliability, maintainability, performance, and accuracy are included

Continued on next page

Audit Checklists, Continued

Definition Phase, (continued)	
Goals (continued)	• Ensure the traceability of requirements from documents • Verify that requirements provide a sufficient base for SW design • Verify that requirements are measurable, consistent, and testable
Product to be Inspected	• User Requirements Document (Trade) • Data Format Requirements • Security Plan
Design Phase	
Goals	• Validate all interfaces among modules within each component • Review list of modules and general function of each module • Validate fault detection, identification, and recovery requirements • Verify component structure meets requirements • Ensure the design meets approved requirements • Validate the selection of reusable components • Ensure traceability of the design to the approved requirements • Validate input and output interfaces
Product to be Inspected	• Functional Requirements • System and Database Design Document • User Requirements
Programming Phase -- Detailed Design	
Goals	• Ensure the design meets approved requirements • Ensure traceability of the design to the approved requirements • Validate all logic algorithms, data structures, and calls within a module

Continued on next page

Audit Checklists, Continued

Programming Phase -- Detailed Design, (continued)	
Goals (continued)	• Ensure that all applicable programming standards are followed • Ensure detailed design meets requirements and is traceable to the System Design document
Product to be Inspected	• System and Database Design Document (Updated) • User Requirements (Intra-agency and Trade updated) • Functional Requirements (Updated) • System Test Plan • Security Plan (Updated)
Programming Phase -- Coding	
Goals	• Ensure the code meets approved standards • Verify technical accuracy and completeness of the code with respect to requirements • Verify code implements the detailed design specifications • Ensure traceability of the code to the approved specifications • Ensure program specifications and code meets requirements and are traceable to the System Design document
Products to be Inspected	• Source Code Listings • Program Specifications • Turnover Package

Continued on next page

Audit Checklists, Continued

System Testing Phase	
Goals	Ensure that : • Test Procedures are written • Each executable statement is executed • Each branch of each decision is executed • All formats and menu options executed • All data types tested • Test cases verify requirements • Test cases verify problem report fixes • Performance tests are met • Regression tests are executed • Stress tests are executed • Anomaly tests are executed (All error messages checked)
Products to be Inspected	• System Test Procedures • System Test Plan (Updated) • Operator’s Manual (if available) • Training Materials • User Guides

Inspection Checklists

Introduction

These Inspection Checklists may be used both while developing the specified work product and as part of technical reviews and deliverable QA reviews.

Software Requirements Inspection Checklist		
#	**Description**	**Comments**
	Accuracy	
1.	Are the requirements correct?	
2.	Will they provide the desired results?	
3.	Are assumptions about system architecture, hardware, and software avoided?	
	Ambiguity	
1.	Can the information be interpreted differently? If so, how?	
	Completeness	
1.	Do the requirements satisfy operational concept and system capabilities?	
2.	Are all interfaces, assumptions, limitations, and constraints identified?	
3.	Are there exceptions which would make the requirements untrue?	
4.	Is the software product described in terms of accuracy, timing, response,/recovery times, throughput, rates, interface control, valid and invalid inputs/outputs, security, and error recovery?	
5.	Are all requirements decisions recorded?	
6.	Have all critical requirements analysis issues been resolved?	
7.	Are the customer and user needs identified and satisfied?	

Continued on next page

Inspection Checklists, Continued

Software Requirements Inspection Checklist (continued)		
#	**Description**	**Comments**
	Complexity	
1.	Do the requirement statements contain more than one requirement?	
2.	Are the requirements too complex to be transformed into logic?	
	Consistency	
1.	Do any of the designated requirements contradict each other?	
2.	Is the information stated with appropriate level of detail?	
3.	Are the requirements consistent with the interface subsystems?	
4.	Is the terminology consistent with the technology of the subject matter?	
	Extraneous	
1.	Are there any requirements which are not relevant to the scope of the problem or solution?	
2.	Do the requirements contain design specifications, schedule, budget, and organizational information?	
	Feasibility	
1.	Can each requirement be implemented with available or expected near term technology?	
2.	Are the appropriate metrics used?	

Continued on next page

Inspection Checklists, Continued

Software Requirements Inspection Checklist (continued)		
#	**Description**	**Comments**
	Flexibility	
1.	Do the requirement accommodate adaptability and reuse without sacrificing performance?	
2.	Are the requirements expressed so that each item may be changed without excessive impact on other items?	
	Redundancy	
1.	Are there any requirements which are stated more than once?	
	Standards	
1.	Do the requirements, diagrams and documentation meet project and customer standards for reuse, maintainability, portability, security, and performance?	
2.	Does the “shall” statement always reference CSCI?	
	Testability	
1.	Can an objective and feasible test with acceptance criteria be constructed to determine whether the requirement is met by the software or system?	
2.	Is it clear how each requirement will be tested throughout the life cycle (i.e., unit, integration, system)?	
3.	Are the requirements quantifiable/measurable?	
4.	Are all inputs and outputs specified?	

Continued on next page

Inspection Checklists, Continued

Software Requirements Inspection Checklist (continued)		
#	**Description**	**Comments**
	Traceability	
1.	Do the requirements correspond to input documents?	
2.	Do the requirements correspond to output documents?	
	Performance	
1.	Have all performance requirements been properly allocated?	

Continued on next page

Inspection Checklists, Continued

Preliminary Design Inspection Checklist		
#	**Description**	**Comments**
	Design	
1.	Does the design comply with the operational concept?	
2.	Does the design provide all of the functional capabilities defined in the requirements specification ?	
3.	Are all requirements addressed in the design?	
4.	Does the design take advantage of any existing software which may be reused?	
5.	Is the design flexible enough to allow for anticipated and unanticipated changes?	
6.	Have all design issues identified in the requirements specifications been recorded and resolved?	
7.	Do any of the concurrent processes in the design create a potential deadlock?	
8.	Which assumptions are valid? Are there any assumptions that can safely be made about the subproduct? Are any assumptions missing?	
	Interface	
1.	Is the prescribed interface correct?	
2.	Does the design perform the specified function?	
3.	Is the user interface relevant and easy to use?	

Continued on next page

Inspection Checklists, Continued

Preliminary Design Inspection Checklist (continued)		
#	**Description**	**Comments**
	Logic	
1.	Does the logic flow in the right direction?	
	Performance	
1.	Does the design impair the performance in any significant capacity?	
2.	Will the design structure function in the time restraints placed on it?	
3.	Do the final performance results indicate that the software sizing and timing allocations are not attainable with the defined architecture?	
	Requirements	
1.	Are the current requirements correct? Are there open action items that might affect the design?	
2.	Are there missing requirements?	
3.	Do the capabilities of the preliminary design structure correspond to current capabilities? Are there any requirements that cannot be traced to design structure?	
	Standard	
1.	Do the preliminary design models and documentation meet project standards?	

Continued on next page

Inspection Checklists, Continued

Detailed Design Inspection Checklist		
#	**Description**	**Comments**
	Data	
1.	Does the design require that all identified data be stored or set prior to its usage?	
2.	Are constant values correct?	
3.	Are there data declarations that will probably never be used?	
4.	Are all data defined and identified?	
	Design	
1.	Does the design provide all of the structures needed to satisfy the functional capabilities defined in the requirements specifications?	
2.	Are processing priorities assigned correctly?	
3.	Are concurrent process, task, communications, synchronization mechanisms correct?	
4.	Are the design structures packaged to promote information hiding?	
5.	Does the design take advantage of existing software that may be reused?	
6.	Is the design flexible enough to allow for anticipated and unanticipated changes?	
7.	Does the design have sufficient level of detail to begin construction?	

Continued on next page

Inspection Checklists, Continued

Detailed Design Inspection Checklist (continued)		
#	**Description**	**Comments**
8.	Have all of the implementation issues that were identified in the requirements specifications been resolved?	
9.	Is the physical database structure complete, consistent with the detailed design, and traceable to a logical scheme?	
10.	Which assumptions are valid? Are there additional assumptions that can safely be made about the subproduct?	
11.	Are all exceptions appropriate and documented?	
	Interface	
1.	Is the prescribed interface correct?	
2.	Does the design perform the specified function?	
	Logic	
1.	Does the logic flow in the right direction?	
	Performance	
1.	Will sizing and timing allocations be attainable with the defined architecture?	
2.	Will the designed software modules meet their performance requirements with the current design?	
	Requirements	
1.	Are the current requirements correct?	
2.	Are there missing requirements?	
3.	Are there any open items that may affect the design?	

Continued on next page

Inspection Checklists, Continued

Detailed Design Inspection Checklist (continued)		
#	**Description**	**Comments**
	Return Code Messages	
1.	Is the error detection design scheme complete, adequate, and consistent?	
	Standard	
1.	Do the detailed design products and documentation meet project and customer standards?	

Continued on next page

Inspection Checklists, Continued

Code Inspection Checklist		
#	**Description**	**Comments**
	Data	
1.	Are any variables used before they are initialized or set?	
2.	Are appropriate "exclusion" mechanisms being used when accessing shared or global data?	
3.	Is the data stored in the correct field of an array?	
4.	Are correct data structures being used?	
5.	Are constant values correct?	
6.	Are there any data declarations that will probably never be used?	
7.	Are the correct data types being used (i.e., character, integer)?	
8.	Are there embedded tables? If so, are they designed for ease of extension (no constraints on size)?	
	Module Design	
1.	Does the code accurately reflect the design?	
2.	Does the code perform only one function?	
3.	Does the function contain a reasonable number of lines?	
	Interface	
1.	Are all parameters necessary?	
2.	Are any parameters missing?	
3.	Are parameters within the design structure consistent, complete, and correct?	

Continued on next page

Inspection Checklists, Continued

Code Inspection Checklist (continued)		
#	**Description**	**Comments**
	Logic	
1.	Does the code perform the required capabilities?	
2.	Are output/input error conditions handled properly?	
3.	Has the entire design been implemented?	
4.	If there are loops, do they execute and terminate properly?	
	Performance	
1.	Does the implementation impair the performance of this software in any significant capacity?	
	Prologue/Comments	
1.	Are the comments meaningful and accurate?	
2.	Are special cases/conditions described?	
3.	Are references to algorithms documented?	
4.	Is the specification portion clear and well documented?	
5.	Does the specification include a description of all algorithmic and control parameters?	
6.	Is information about interfaces (if any) to other functions described?	
	Requirements	
1.	Do all functions of the code correspond to current requirements?	
2.	Are the current requirements correct? Has a requirement change impact analysis been performed for differences?	

Continued on next page

Inspection Checklists, Continued

Code Inspection Checklist (continued)		
#	**Description**	**Comments**
3.	Are there open action items which might affect the code?	
	Return Code Messages	
1.	Are all exit conditions detected and reported?	
2.	On exits should a return code be set or message issued?	
	Standards	
1.	Does the code meet project and customer standards?	
2.	Is the program modification log updated to clearly describe all changes to production code?	
	Syntax	
1.	Are there statement delimiters on every line?	
2.	Are all types defined?	
	Testability	
1.	Can each function be tested individually?	
2.	Can the test data be reused? Is there a storage or library for test data?	
3.	Are there test tools available? Is there a library of test tools/utilities?	
4.	Are test cases defined with expected input and output?	
	Portability	
1.	Is the code built such that it may accommodate change with relative ease?	
2.	Are DBMS-dependent operations isolated?	

Continued on next page

Inspection Checklists, Continued

Test Plan Inspection Checklist		
#	**Description**	**Comments**
	Ambiguity	
1.	Are the plans for installing, controlling, and maintaining items in the software test environment clear?	
	Completeness	
1.	Is sufficient test support planned for?	
2.	Do tests cover quality objectives?	
3.	Have test cases been identified in the plan which exercise combinations of different functions and objects simultaneously?	
4.	Are test cases planned to validate user and maintenance documentation?	
5.	Have test cases been set up to verify valid and invalid data?	
6.	Are test cases set up for verifying error and exit conditions?	
7.	Does each test case specify inputs, equipment, environment, procedures and expected outcome?	
8.	Is the test case adequate to verify requirements?	
	Consistency	
1.	Is the scheduled test sequence logical?	
	Correctness	
1.	Is the traceability matrix between the tests and the requirements accurate?	

Continued on next page

Inspection Checklists, Continued

Test Plan Inspection Checklist (continued)		
#	**Description**	**Comments**
2.	Are the correct tests being used to test each function or object?	
3.	Are test estimates based on historical data as much as possible?	
	Requirements	
1.	Is there any unnecessary redundancy between tests?	
2.	Are all allocated requirements covered?	
	Standards	
1.	Does the test planning conform to project and customer standards?	
	Traceability	
1.	Are the test cases fully traceable to the software requirements?	
2.	Are all of the software requirements addressed by at least one test case?	
3.	Do the tests verify all user, hardware, modules, database, and other product/ system interfaces?	

SDLC Phase Checklists

Introduction

One of the most reliable and simple ways of determining if everything has been done, that should be done, is to use checklists. The checklists included in this appendix are presented in the order of phase/products.

Checklists are frequently used after completing an important milestone; however reviewing the appropriate checklist(s) before proceeding to the next phase can serve as a useful reminder about all of the issues to be addressed. Action may not be required for some items. It is recommended that these checklists be reviewed before performing each phase's quality gate review.

In addition to checklists and milestone reviews, conducting quality assurance reviews and preparing the system test plan for approval are important factors in verifying and validating the software planning.

In This Subsection

Initiation Phase Checklist

Pre-IRB, Project Initiation

- Have the legislative mandates to which this project must comply been identified? *This includes requirements for the Customs Modernization Act and/or audits requirements that are driving the project.*

- Do the proposed solutions focus on, and support OIT goals and objectives as stated in the current Annual Plan (or will those objectives be changed)?

- Has an explanation been provided on how Customs mission or strategic plans are supported by this project?

- Has the impact of not implementing this project been stated? Have you rated the impact of non-implementation, e.g. low, medium, or high?

- Are cross functionality benefits provided? Have beneficiaries of the project and opportunities to leverage this project across business organizations been explained?

- Have all Business Process Improvements (BPI) associated with this area or function of business been considered in the solution?

- Has a Cost/Benefit Analysis (CBA) document been prepared to provide adequate cost and benefit information?

- Has the process through which costs were estimated been defined?

- Are the cost estimates supported?

- If COTS are being used, have all customization costs been provided?

- Does the CBA specify the analysis and evaluation of alternative approaches, provide proposed solutions, and list the benefits attained through the development of each alternative?

- Does the CBA include a comparative cost/benefit summary?

- Is a copy of the preliminary cost/benefit analysis for this project attached?

Continued on next page

Initiation Phase Checklist, Continued

Pre-IRB, Project Initiation (continued)

- Have areas of risk been identified and mitigating factors for each risk stated?

- Have organizational risks been identified with detail information on which parts of the organizations will be impacted? Is information also included on: whether this project will cause a significant organizational change, process redesign, and/or a change to the way peoples' jobs are performed? Are all proactive measures taken to mitigate this risk included?

- Does the proposed system architecture conform to USCS standards? Is an explanation provided if there is any deviation?

- Is there a proposed acquisition strategy?

- Has mission effectiveness been supported with information on: identification of stakeholders (functional areas) serviced by the proposed project, the reasons why the recipient considers the project to serve a critical function their operation, and has each service been identified?

- Has the mission effectiveness for all external stakeholders been stated along with the criticality of the function being supported or improved? Has information on the service to each of the external stakeholders been provided?

Continued on next page

Initiation Phase Checklist, Continued

The Project Plan

- Have the project organization and responsibilities been defined?
- Has the work breakdown structure (WBS) been used as the basis for budget development, allocation and planning?
- Has planning been based on the contract schedule, budgets, user requirements, Customs SDLC, operational and technical environment, and the personnel and resources?
- Does the Project Plan Schedule show activity initiation, dependencies on other events, and key development milestones?
- Does the Project Plan list all tasks and activities that the user must be involved in?
- Are work schedules based on WBS, the plans, the budget, and technical parameters?
- Is the software development methodology that will be used for the project documented in the project file and are the required deliverables clearly defined?
- When the Customs SDLC Life Cycle and methodology has been tailored, is the tailored life cycle documented in the project file? Are the reasons for tailoring documented?
- Does the Project Plan provide for the review and audit of activities such as: estimating and planning, reviewing, and maintaining project commitments?
- Have management and user approvals to proceed with the project development been obtained?
- Has the Project Plan been approved by senior management?
- Has the Project Plan been baselined and entered into the project file/library?

Continued on next page

Initiation Phase Checklist, Continued

Project Plan: Security Checks

- Are security and data access requirements identified and described?
- Have security considerations been proposed for the protection of the:
  - Facility
  - Environment
  - Personnel
  - Data
- Is the proposed system covered by the Privacy Act? If so, is a Privacy Act Notice being drafted?
- Has a process been established to address privacy issues having an impact on the proposed information system?
- Have project team members been scheduled for training in security awareness and privacy issues relative to the project?
- Does the magnitude of the information system development effort require a privacy plan?
- Have provisions been made for required disclosure accounting?

Continued on next page

Initiation Phase Checklist, Continued

Project Plan: Coordination & Control Activities

- Have all groups that are impacted by software sizing estimates such as effort and cost, schedules and any other commitments had an opportunity to review these planning estimates?

- Has the Project Manager negotiated all project commitments?

  Example: Commitments are negotiated between the Business Sponsor, Project Manager, the Project Software Manager and all other project Team Leads. Commitments that have been agreed to must be documented.

- Has the **involvement of other affected groups** for the software development activities been negotiated and documented?

  Example: Commitments and schedules are negotiated between the Business Sponsor, Project Manager, and all other affected support organizations such as the Data Administration and Database Management Teams, the AIS Security Team, the Training/Documentation Team, the SAT Team, and the Users. Commitments that have been agreed to must be documented.

  Reference: Volume II, Chapter 17, Section B, *Functional Impact Areas Checklist*, for potential affected functional groups.

- Are **changes to software commitments** being made with the involvement and agreement of all affected groups?

  Example: Commitments and change priorities and impacts are negotiated between the Business Sponsor, Project Manager, and the project's Change Control Board; impacts of changes to schedules and delivery agreements are communicated and renegotiated with the System Development Team and all affected support organizations. Commitments that have been agreed to must be documented.

Continued on next page

Initiation Phase Checklist, Continued

Project Tracking

- Is a documented Software Development Plan (i.e., the overall Project Plan and other associated planning documents) being used for tracking the software activities and communicating project status?

- Are all of the software development estimates documented for use in planning and tracking the software project?

- Are specific procedures defined for monitoring project progress and reporting?

- Are all of the actual project results/performance tracked against the Project Plan? **Note:** Actual performance is reviewed during every life cycle phase.

- Is the project's effort and cost being tracked, and corrective actions taken as necessary (during every life cycle phase)?

- Are corrective actions taken when the actual results and performance deviate significantly from the Project Plan?

- Is the actual measurement data and replanning data for the software project recorded?

- Were reviews and audits of the following activities conducted?:

  - Those activities that review and revise commitments

  - Activities for revising the Software Development Plan

  - Activities that develop the content of the Software Development Plan

  - Activities that track cost, schedule, risks, technical and design constraints, functionality and performance, and also the activities for conducting planned technical and management reviews

Continued on next page

Initiation Phase Checklist, Continued

Project Plan: Quality Assurance

- Does the Project Plan define how Risk Management will be performed?
- Have the quality factors that are most important for the system been identified?
- Are the management and technical controls to be applied to the project identified in the Project Plan?
- Does the Project Plan provide for the following quality assurance reviews:
    - **Product evaluations** - the Project Plan lists the products to be evaluated and the evaluation criteria
    - **Management reviews** - formal evaluation of project level plans or project status relative to these plans
    - **Technical reviews** - identify discrepancies from specifications and standards
    - **Software inspections** - to detect and identify defects
    - **Configuration audits** - functional and physical
    - **Walkthroughs** and Peer Reviews
    - **SDLC Compliance audits**
    - **Corrective Action System** - document problems, implement a closed loop corrective action system, track status of all problems to closure

Continued on next page

Initiation Phase Checklist, Continued

User Requirements

- Was a comprehensive analysis performed on the current system?
- Has a User Requirements Document (URD) been prepared that defines the problem to be addressed by the project?
- Does the URD include the objectives to be met by the new system?
- Does the URD include a description of current methods and procedures, both manual and automated?
- Does the URD include a description of the proposed solutions including the information needs?
- Does the URD state the acceptance requirements for the system?
- Were the allocated user requirements reviewed by the project managers, project software managers, team leads and other groups such as system analysts, database, system testing?
- Are all Requirements baselined?
    - Hardware Requirements
    - System Requirements
    - Functional Requirements
- Has the user signed off on the allocated requirements to be baselined?
- When requirements change, are the project plans, products and activities revised; and are changes to commitments re-negotiated?

Definition Phase Checklist

Functional Requirements

- Does the Functional Requirement Document (FRD) describe how the functions and subfunctions required by the user will be reflected in the system architecture?
- Does the FRD have a statement of objectives to be met by the system?
- Does the FRD state all alternative solutions that were proposed?
- Does the FRD identify all assumptions and constraints?
- Does the FRD describe the inputs and outputs of the system?
- Is there a description of system software needed to support the system?
- Is there a description of interfaces with other systems?
- Is there a description of the system sizing and capacity requirements?
- Does the FRD describe the characteristics of each data element, the retention and the ***data security requirements**?*
- Does the FRD explain the ***security protection requirements*** of the project, and identify privacy issues?
- Does the FRD describe the performance requirements of the application system?
- Does the FRD define a procedure when there are system failures and state the alternative courses of action?
- Are all of the functions traceable to the User Requirements document?
- Has a requirements traceability matrix been completed?
- When requirements change, are the project plans, products and activities revised; and are changes to commitments re-negotiated?

Continued on next page

Definition Phase Checklist, Continued

Data Administration During Definition

The project scope and mission requirements are to be reviewed at the start of each phase to determine if the information needs are still valid. The Data Management Plan begun in this phase is, like the Security Plan, a continually growing document throughout the life cycle. The following questions should also be referred to during those later phases.

- Has the Project Manager selected a Data Administration Liaison and a secondary backup from the project team to work with the Data Administration Branch?
- Have the roles and responsibilities been defined in the Data Management Plan for each team member that will be involved in developing the information requirements?
- Has a copy of this plan been provided to the Data Administration Branch?
- Has a list of all entities been developed? For each entity, is there a precise business definition along with the key attributes which uniquely identify an occurrence of the entity?
- Have Conceptual/Logical Data Models been developed? These models for a specific project show the entities and their interactions or relationships.
- Have Definition reports for logical/physical structures been generated?
- Have all likely organizations that supply data to the application been identified?
- Has a plan for the physical flow of information through the new system been developed?
- Did the team validate that each entity in the high level information flow is in the conceptual model and vice versa?
- Is the usage of each entity shown in matrices by function and/or by distribution/location?

Continued on next page

Definition Phase Checklist, Continued

Risk Management

- Have tangible project risks been identified?
- Have Mitigation Plans been developed to describe, measure and proactively manage each risk?
- Has each risk item been evaluated for its importance to the project?
- Has a safeguard measure been implemented to mitigate the impact of each risk?
- Were all project risks tracked for progress?
- Was corrective action taken to reduce the negative impact of any specific project risk?

System/Security Test Plans

- Does the System Test Plan (STP) specify that at least one user representative will participate in the testing?
- Does the STP include detailed descriptions, procedures, and anticipated results for each test?
- Does the STP have sufficient volumes of test transactions which have a wide range of valid and invalid conditions?
- Does the STP provide for the training of employees that will conduct the testing?
- Does the STP provide for program, subsystem and entire system testing?
- Does the STP test for transfer of data between other systems?
- Does the STP require regression testing, parallel testing, performance testing, and graphic user interface testing when applicable?
- Does the STP call for compatibility testing with other hardware, software, and NetWare and also applications running in the environment?

Continued on next page

Definition Phase Checklist, Continued

System/Security Test Plans (continued)

- Does the STP include locations, dates for testing, and milestone events?
- Does the STP include equipment, software, and personnel resources needed for testing?
- Does the STP include criteria used to evaluate the test results and record them?
- Does the STP identify the constraints, if any, that would affect the testing or test results?
- Does the STP include test descriptions for each test, e.g. controls, inputs, outputs, procedures, etc.?
- Does the STP include information for the required Security Tests or is that defined in a separate document? If a separate Security Test Plan is defined, have all the above considerations been checked against that document also?

Security Plan

- Has the system environment for the application and planned inputs, outputs and interfaces been identified and documented?
- Have security considerations and risks been defined and documented?
- Have security controls over management and technical data for the development project itself been planned and documented?
- Have the data security requirements been identified and documented in a plan which addresses how the security requirements will be implemented?
- Does the Security Plan identify the sensitive data for the application and provide the measures that will be taken to protect the data?
- Do the security controls for protecting the data meet the requirements as defined by the AIS Security Team, Treasury Security Manual 71-10, and the Risk Assessment Guideline TD P 85-03?

Continued on next page

Definition Phase Checklist, Continued

Trusted Facility Manual

- Does the Trusted Facility Manual (TFM) document the threats, countermeasures, vulnerabilities and protection mechanisms?
- Does the TFM identify the audit mechanisms?
- Does the TFM include all commands, system calls, and function definitions to be encountered at the facility?

Disaster Recovery/ Contingency Plan

- Has a Disaster Recovery/Contingency Plan been drafted? Has it been updated after the Design Phase is completed?
- Does the Disaster Recovery/Contingency Plan contain procedures to follow before and after normal processing interruptions, as well as scheduled testing frequencies for the plan itself?

Training Plan

- Has a Training Plan been developed; and has it been approved by the User?
- Have target audiences for training been identified and a training needs analysis performed?
- Are choices of instructional methods and media for effective training justified by the subject/course objectives, audience, and constraints imposed?
- Have sufficient materials, resources, and time been allocated to specified courses to ensure appropriate instruction for each target audience group?
- Are specific courses described in enough detail to permit evaluation of the planned training schedule, objectives, training material, required equipment and resources, and administrative procedures to be used?
- Have methods been defined for maintaining quality control over course development and effectiveness?
- Do documented procedures exist that explain the use and development of a training database if required?
- Have plans been specified for training follow-on personnel and updating courses as required?

Design Phase Checklist

System Design Checks

- Are the performance requirements of the system adequately defined?
- Is all the necessary system software defined?
- Have all security and Privacy Act requirements of the system been met?
- Has the final Privacy Act Notification been prepared and approved and forwarded to be announced in the Federal Register?
- Is the operating environment defined?
- Are hardware, software, and services acquisitions planned?
- Has the logical database been approved?
- Are data management considerations defined?
- Have system acceptance criteria been documented?
- Have operating backup facilities been addressed?
- Has the security design been documented and approved?
- Have detailed system/subsystem specifications been developed for the system?
- Do the detailed system/subsystem specification include the design characteristics of the system and provide a system flowchart?
- Does the detailed system/subsystem specifications include interfaces to other systems?

Continued on next page

Design Phase Checklist, Continued

System Design Checks (continued)

- Have manual procedures and support measures been documented?
- Does the data repository contain all data element names, attributes, validation rules and definitions?
- Have all data elements been defined and their usage identified? Have master files/databases been defined?
- Are definitions of inputs from and outputs to interfacing system referenced?
- Have edit criteria and transaction conditions been established and defined for online transactions?
- Have telecommunications requirements been fully documented?

Data Administration During Design

- Have Interview Plans been developed which document the relationship between the data requirements interview and the functional requirements interviews?
- Have the information requirements been analyzed for each process identified during the functional requirements?
- Has the methodology used to normalize the entities required by each process been documented?
- Are all data items compliant with the current ANSI SQL Standards?
- Has the Conceptual Data Model been kept up-to-date whenever new entities are identified? Are data entities representing all the data in the data flows also in the model?

Continued on next page

Design Phase Checklist, Continued

Data Administration During Design (continued)

- Has the Project Team validated the logical data model for completeness of all data elements in the flows?
- Have the Users reviewed the logical data model for accuracy?
- Has the Data Management Plan been documented to validate that all data elements in the flow diagrams are included in the logical data model?
- Have data retention requirements been reported for each stage in the archival/migration process of the data?
- Have the data security requirements been identified and documented in a plan which addresses how the security requirements will be implemented?
- Does the Security Plan identify the sensitive data for the application and provide the measures that will be taken to protect the data?
- Do the security controls for protecting the data meet the requirements as defined by the AIS Security Team, Treasury Security Manual 71-10, and the Risk Assessment Guideline TD P 85-03?
- Have all the latest changes to the logical data model been applied?
- Has the physical database design structure been completed?
- Does the Corporate Data Dictionary contain the definitions and usage of data for this application?
- Has the Data Administration Liaison provided complete, precise, and accurate information to the Data Administration Branch as needed?
- When data conversion is required, is there a Conversion Plan documented?

Continued on next page

Design Phase Checklist, Continued

Detailed Design Checks

- Does the system structure clearly show boundaries of each design unit or job stream?
- Has the physical database/file design been documented?
- Have teleprocessing specifications been completed?
- Does the data repository contain all data flows and data stores used throughout the system?
- Does the System [Detailed] Design document include detailed system analysis and design?
- Have the input record formats and descriptions been provided?
- Have detailed program specifications been developed for all programs of the system?
- Do these specifications include a narrative description of the program and its functions?
- Are program performance requirements described?
- Have all program controls been described?
- Is there a description of the program's logic, including flowcharts and decision tables, supplemented with narrative explanations?
- Have the logical and physical characteristics of all databases used by the program, including file layouts and data element definitions, been provided?
- Have the database specifications been included?
- Has the I/O design been fully documented?
- Are all interactive dialogues fully documented?
- Are all functional requirements traceable to the design document(s)?

Programming Phase Checklist

Checks Before Coding

- Is a conversion effort planned and documented?
- Is there a clear separation and definition of:
  - Permanent and temporary files/tables?
  - Input and output files/tables?
  - Mainframe, server, and PC files/tables?
- Does the data repository contain all data flows and data stores used throughout the system?
- Has each requirement in the functional requirement document been identified using a traceability matrix?
- Was a design walkthrough conducted and was permission granted to begin programming?
- Do the specifications describe the application performance requirements?
- Is the description of the program's logic, including flowcharts and decision tables, supplemented by narrative explanations?
- Are the logical and physical characteristics of all databases used by the programs provided? File layouts and data element definitions should also be provided.

Checks After Coding

- Have job control commands for integration test, acceptance test, and the production environment been prepared and validated?
- Are commercial software products to be tested concurrently?
- Have security functions been performed according to project specifications?
- When errors are found in a particular module, are there efforts to find other, perhaps related or subsequent errors in the same module or segment?
- Is the test document complete and well organized?

Acceptance Phase Checklist

Data Administration After Programming

- When there are additional data backup, logging, recovery, and disaster recovery requirements for this application other than for production software; then have these requirements been provided to Operations?
- Have all data management concerns been documented in the appropriate deliverables not just in the model/dictionaries?
- Do the user training materials include information on data creation, data collection, data validation and quality assurance issues?

Data Administration During Acceptance

- Have the data testing strategies been documented in the Test Plan? Do these strategies include testing for database performance and functionality?
- Does the Test Plan includes the strategy for supporting integration and acceptance testing of the databases?

Concerning the User Acceptance Checklist

The following checklist is designed for the User of the system and spans items that may have been performed and completed in earlier phases. It is essential that this checklist be referred to during Project Planning.

The Project Manager must require all Team Leaders to work with the user of the system while preparing the criteria for the software inspections.

User Acceptance

- The person who will sign off on the entire system (i.e., the Business Sponsor/Process Owner) is involved in previous critical reviews and quality gate activities, particularly those involving the functionality of the system. If this responsibility is delegated, has it been delegated to a person with the authority to approve the system?
- User Acceptance is a gradual process. Has the user had an opportunity to review the system components as they were completed or at a minimum at the end of each phase? Has the user also had opportunity to review and approve the user documentation and training materials?
- Has the user been involved in how the system looks and behaves, i.e. screen interface design?

Continued on next page

Acceptance Phase Checklist, Continued

User Acceptance (continued)

- The user acceptance criteria is finalized during the definition and design phase. Have these criteria been used to validate the product results?

- The acceptance criteria should not be ambiguous, but objective and measurable. Are the expected results very precisely defined so that testing can be designed to prove that the correct result has been achieved?

- Has the user had opportunity to review the planning/acceptance documents:

  - Requirements Documents
  - Design Documents
  - Test Plans
  - Test Data

- System testing is the culmination of the design effort and a critical component of final acceptance. Therefore, does the System Acceptance Test (SAT) Team include at least one active user participant?

Implementation Phase Checklist

System Implementation Checks

- Have personnel requirements, staffing and training requirements for the implementation been identified?
- Has the validated version of the software and documentation been provided for implementation?
- Are points of contact with phone numbers listed for the specific types of problems that may occur?
- Have preparations and notifications been made to migrate to the production environment?
- Has the new application or enhancement been included in the Disaster Recovery/Contingency Plan testing procedures?
- Has an acceptable level of risk been established for the system?
- Has an Accreditation Statement been signed and placed in the project file?
- Did each team member participate in the Lessons Learned review; and do they have copies of the documented project experiences?

Data Administration During Implementation

- Have all cut over activities been determined for Data Administration and Configuration Management to support the migration of the system from one environment to another?
- Has the project team verified that the Data Management Plan and Configuration Management Plan are synchronized and do not contradict each other?
- Have all data management activities been determined for supporting the audit and evaluation process as well as responding to recommendations from the evaluation report?

Operation Phase Checklist

Monitor Performance

- Is the system performing as specified?
- Are problems and recommendations for enhancements being identified?
- Have discovered problems been properly logged and acted upon?
- Is there an approved method for reporting problems?
- Is analysis performed on what types of problems occur and where in the system they occur, with the goals of eliminating causes of these problems and changing development processes to reduce future problems?

Evaluation Phase Checklist

System Evaluations

- Have the selection criteria been established to identify systems for review?
- Has the type of review been decided?

  Examples: Project Evaluation, User Satisfaction, Post-Implementation, Disaster Recovery/Contingency Plan Evaluation, Risk Analysis, SDLC Compliance, IRM Review, Internal Controls, Financial Management.
- Has the scope of the specific review been clearly delineated?
- Have the objectives for the review been determined?
- Has a measurement strategy been determined?
- Has a review plan been completed?
- Has it been determined who will be responsible for addressing any findings and recommendations made in the report?

This page intentionally blank